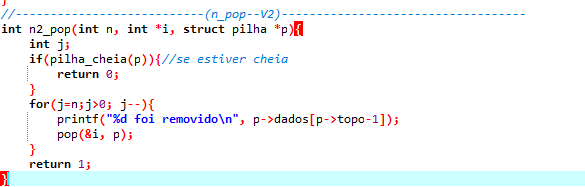
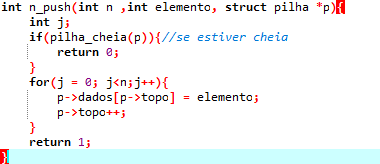
1. **A) utilizando somente as funções definidas, construir uma função n\_pop, que desempilha n elementos da pilha. Faça todas as verificações necessárias e retorne resposta adequada ao usuário.**
2. int n2\_pop(int n, int \*i, struct pilha \*p){
3. int j;
4. if(pilha\_cheia(p)){//se estiver cheia
5. return 0;
6. }

for(j=n;j>0; j--){

1. printf("%d foi removido\n", p->dados[p->topo-1]);
2. pop(&i, p);
3. }
4. return 1;
5. }

1. **B) utilizando somente as funções definidas, construir uma função n\_push, que empilha o elemento recebido n vezes. Faça todas as verificações necessárias e retorne resposta adequada ao usuário.**
2. int n\_push(int n ,int elemento, struct pilha \*p){
3. int j;
4. if(pilha\_cheia(p)){//se estiver cheia
5. return 0;
6. }
7. for(j = 0; j<n; j++){
8. p->dados[p->topo] = elemento;
9. p->topo++;
10. }
11. return 1;
12. }

1. d) Construir uma função que recebe uma pilha **p** e um inteiro **n**, remove todas as ocorrências desse inteiro, se ele estiver na pilha. A função devolve 1 em caso de sucesso ou zero em caso de fracasso (não existir o valor **n** na pilha). Não se esqueça de preservar a pilha, a menos dos valores retirados. Você pode usar pilhas auxiliares na sua função. Utilize as funções de manipulação.

int limpa\_x(struct pilha \* p, int x){

int i, cont = 0;

struct pilha \* aux = inicia\_pilha(p->topo);

//printf("\nCapacidade da pilha: %d\n", p->topo);

if(pilha\_vazia(p)){

return 0;

}

while (!pilha\_vazia(p)){

pop(&i, p);

if(i!=x){

push(i, aux);

}

else{

cont++;

}

}

while (!pilha\_vazia(aux)){

pop(&i, aux);

push(i, p);

}

if(!cont){

return 0;

}

return 1;

}

Construir uma função para testar se uma pilha P1 tem mais elementos que uma pilha P2. Considere que P1 e P2 já existem.

int compara\_elemento\_pilha(struct pilha \* p1, struct pilha \*p2){

int i=0, j=0;

while (i<p1->topo){

i++;

}

while (j<p2->topo){

j++;

}

printf("Iqt = %d, Jqt = %d\n", i, j);

if(i==j){

printf("mesma quantidade de elementos\n");

return 1;

}

else{

printf("quantidade de elementos diferentes\n");

return 0;

}

}